IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant: Hugo Cheung
Art Unit: 2182
Serial No.: 10/650,403
Examiner: Tanh Q. Nguyen
Filed: 08/28/03
Docket: TI-32740.1
For: SERIAL PERIPHERAL INTERFACE WITH HIGH PERFORMANCE BUFFERING SCHEME

APPELLANTS' BRIEF UNDER 37 C.F.R. §1.192

Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Commissioner:

The following Appeal Brief is respectfully submitted in connection with the above-identified application in response to the Final Rejection mailed June 23, 2006. Please charge all required fees, including any extension of time fees, to the deposit account of Texas Instruments Incorporated, Deposit Account No. 20-0668.

REAL PARTY IN INTEREST

The real party in interest is Texas Instruments Incorporated, to whom this application is assigned.

RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences known to the Applicant's legal representative.

STATUS OF THE CLAIMS

Claims 15-17, 19, and 20 are the subject of this appeal. Claims 15-17, 19, and 20 are rejected. This application was filed on August 28, 2003.

STATUS OF THE AMENDMENTS

The Appellants filed an amendment under 37 C.F.R. § 1.116 on September 25, 2006 in response to the Office Action dated June 23, 2006. The amendment was not entered.

SUMMARY OF CLAIMED SUBJECT MATTER

Specification page 9, line 3 to page 19, line 13, provides a concise explanation of the invention defined in independent claim 15.

With reference to Figure 3, an exemplary SPI configuration 300 is illustrated. In accordance with this exemplary embodiment, SPI configuration 300 comprises an SPI configured in a master/slave arrangement, i.e., SPI configuration 300 can be programmed in master mode or slave mode. For example, an exemplary SPI configuration 300 can be suitably implemented within a master device, such as a microprocessor, or a slave device, such as an A/D converter.

In addition, SPI configuration 300 suitably comprises a single buffer configuration 301 configured for facilitating the providing of a high data rate without increasing the logic area. In accordance with the exemplary embodiment, single buffer 301 comprises a transmit shift register (Tx Shift Reg) 302 and a receive shift register (Rx Shift Reg) 304. In addition, single buffer 301 can be configured in a queuing arrangement, for example, a FIFO buffer arrangement 308. In accordance with this exemplary embodiment, FIFO buffer arrangement 308 can comprise various sizes, for example, from FIFO (0) to FIFO (N-1), where N = 2 through 128 or more, i.e., a FIFO buffer can be configured for a data storage area of 2 bytes to 128 bytes or more.

As discussed above, the single buffer scheme for the SPI configuration 300 can provide high data rates with only half of the size of double buffering schemes, approximately double the data rate using the same buffer area as double buffering schemes, or an increased data rate using less buffer area than double buffering schemes. In accordance with another exemplary embodiment of the present invention, an exemplary SPI buffering scheme includes queuing configuration comprising a counter/pointer arrangement for facilitating the throughput of data through the SPI as well as for reducing the CPU overhead.

In accordance with such an exemplary embodiment, with reference again to Figure 3, single buffer 301 includes an exemplary counter/pointer arrangement 316 comprising three pointers configured for designating the locations in the FIFO buffer 308

for writing and receiving of data. For example, one pointer can be used to indicate the location where the CPU will write the data to be transmitted (WrPtr), another pointer can be used to identify the location where the received data can be stored and read by the CPU (RdPtr), and a third pointer can be used to identify the location where the data to be transmitted is located and the location where the received data will be stored after shifting of the registers is completed (ShfPtr). In addition, counter/pointer arrangement can comprise two counters configured to suitably track the number of bytes of data that have been received (RdShfCnt) and the number of bytes of data that need to be transmitted (WrShfCnt). Moreover, additional or fewer counters and pointers can be implemented depending on any number of design criteria. As will be described in more detail below, the use of the counters and pointers can provide a buffering scheme facilitating high data rates without increasing CPU overhead.

In addition, single buffer 301 can include one or more data registers 314 which are configured with FIFO buffer 308 for the storing of data for further processing and use. Data register 314 is configured to permit data to be written to by a CPU, and for data to be read from by a CPU. Data register 314 can be configured in various sizes to hold various amounts of data, for example, a few bits, a single byte , or several bytes or more.

Still further, single buffer 301 can comprise various other components configured to facilitate operations of SPI 300. For example, single buffer 301 can comprise a bit counter 306 which is configured to suitably track the number of bits transmitted or received. As a result, bit counter 306 can facilitate confirmation by single buffer 301 that a complete transmission or receipt of data to and from another device as been accomplished.

In addition to single buffer 301, SPI configuration 300 can include various other elements to provide a high performance operating system. In accordance with an exemplary embodiment, to facilitate a reduction in the polling of the microprocessor or CPU, SPI configuration 300 can be configured with interrupt signals to indicate to the CPU that data is ready for transmitting, or that buffer locations are available for receiving data. For example, SPI configuration 300 can be configured to suitably

include two interrupt requests, such as a transmit interrupt request (TxIRQ) and a receive interrupt request (RxIRQ). Each of interrupt requests TxIRQ and RxIRQ can be suitably configured to provide a signal informing a master or slave device that SPI configuration 300 is ready to transmit or receive data. In addition, as will be described in more detail below, interrupt requests TxIRQ and RxIRQ can be configured with a threshold value that determines when the CPU should intervene and facilitate the transmitting or receiving of data through SPI configuration 300.

Further, SPI configuration 300 comprises an SPI/GPIO control component 310 configured for communication with another device. Control component 310 suitably includes interface pins which can be configured as general purpose input/output pins (GPIO), for example, an SPI clock (SCLK) pin, a master in/slave out (MISO) pin, a master out/slave in (MOSI)pin, and an SPI select/active low (SSN) pin. Additional pins can also be included within control component 310 for other interfacing purposes.

As a result of queuing configuration within single buffer 301, SPI configuration 300 can provide a high data rate without increasing the logic area as required by prior art buffering schemes. In addition, single buffer 301 can be configured in any queuing arrangement configured to provide a high data rate, for example, circular queuing arrangements, or linear queuing arrangements. To facilitate an explanation of the operation of an exemplary queuing arrangement for single buffer 301, an exemplary circular buffer is illustrated and discussed below.

With reference to Figure 4, an exemplary FIFO buffer scheme 400 having a pointer/counter arrangement is illustrated. In accordance with this exemplary embodiment, the FIFO buffer scheme 400 comprises three pointers, for example, a write pointer (WrPtr), a read pointer (RdPtr), and a shift pointer (ShfPtr), and two counters, for example, a write shift counter (WrShfCnt) and a read shift counter (RdShfCnt). The range of values for these pointers and counters is from 0 to N, where N =2 through 32 or more. In the exemplary embodiment, N=16, i.e., the SPI buffer is configured for 16 byte operation.

In accordance with this exemplary embodiment, the WrPtr pointer comprises the location of the FIFO buffer 400 that the CPU will write to during data transmission, and

is configured to increment after the CPU writes to the SPI data register. The RdPtr pointer comprises the location of the FIFO buffer 400 that the CPU will read from during data transmission, and is configured to increment after the CPU reads from the SPI data register. The ShfPtr pointer comprises the location that the shift registers, for example, the transmit shift register 302 and the receive transmit register 304, will read from and write to when the registers are receiving or transmitting data. In the illustration of Figure 4, WrPtr pointer has a value of 8, RdPtr pointer has a value of 0, and ShfPtr pointer has a value of 3.

As for the counters, the WrShfCnt counter is configured to provide the count number that the WrPtr advances the ShfPtr during transmission of data, e.g., WrShfCnt counter has a value of 00101B (5 decimal) in the illustration of Figure 4, and indicates the number of bytes remaining for transmission. In addition, WrShfCnt counter is configured such that when WrShfCnt counter has a value of N, e.g., 10000B (16 decimal), writing to the SPI data register will not increment the WrShfCnt counter value. As a result, any write operation will operate to overwrite the previous written value.

Meanwhile, the RdShfCnt counter is configured to provide the count number that the ShfPtr advances the RdPtr during receipt of data, e.g., RdShfCnt counter has a value of 00011B (3 decimal) in the illustration of Figure 4, and indicates whether or not that the SPI has received data. In addition, RdShfCnt counter is configured such that when RdShfCnt counter has a value of zero, e.g., 00000B (0 decimal), reading from the SPI data register will not decrement the RdShfCnt counter value. As a result, any read operation will operate to re-read the previous read value.

In accordance with another aspect of the present invention, the buffering scheme is configured for reducing the amount of CPU polling or interrupt overhead required by prior art buffering schemes. As discussed above, prior art buffering schemes typically require high interrupt overhead due to the need to frequently interrupt the CPU. For example, for double buffer schemes, the CPU has to be configured to service both buffers which requires additional interrupt overhead. Further, since such prior art buffers generally do not know when data is ready for transmitting or when data is ready for receiving, these prior art buffers must frequently poll the dual buffers to identify the

appropriate time to transmit or receive data, whether or not the dual buffers are ready to transmit or receive data.

However, through use of the exemplary pointer and counter configuration, FIFO buffer scheme 400 is configured to permit SPI configuration 300 to suitably transmit data until a threshold level of received bytes is left for sending, e.g, three bytes or less. This threshold level can be suitably configured to different levels, e.g., one, two, four or more, or can be completely disabled as desired. Upon reaching the threshold level of available bytes, FIFO buffer 400 can be configured to interrupt the CPU and request additional data. Accordingly, the CPU could again begin to fill FIFO buffer with data to a desired level, e.g., until 16 bytes are received and stored, or any number desired. In addition, once the FIFO buffer is approximately full, for example, only a few registers are available for the filling of data, FIFO buffer scheme 400 can be configured to interrupt the CPU to indicate that data is ready for transmitting.

For example, with reference to Figure 4, if the threshold level of bytes to be received before interrupting the CPU is two bytes, no interruption of the CPU will occur since the RdShfCnt counter equals three, i.e., three bytes have been received. However, if further data is transmitted, and the counters and pointers are properly decremented such that two or less bytes of data are available, FIFO buffer scheme 400 can be suitably configured to interrupt the CPU to request additional data. Moreover, an upper threshold level of bytes can be provided such that when the FIFO buffer is full, or approaching a full buffer status, e.g., one or two bytes of area available for filling of data, FIFO buffer scheme 400 can be configured to interrupt the CPU to request the transmission of data.

Thus, in accordance with an exemplary embodiment, both the WrShfCnt counter and the RdShfCnt counter can be configured to operate with interrupt request signals, for example, the transmit interrupt request (TxIRQ) and the receive interrupt request (RxIRQ), which can provide a signal indicating that SPI buffer scheme is ready to transmit or receive data. For example, in the event that WrShfCnt is less than or equal to the value of transmit interrupt request (TxIRQ), the signal and status flag of the transmit interrupt request (TxIRQ) will be activated. Further, in the event that RdShfCnt

is larger than or equal to the value of receive interrupt request (RxIRQ), the signal and status flag of the receive interrupt request (RxIRQ) will be activated. In addition, the values of the TxIRQ and RxIRQ can be configured at different levels depending on the desired intervention of the CPU.

While it should be noted that the above exemplary embodiment can provide an interrupt feature to prevent excessive polling of the CPU, it should be noted that the CPU can continue to provide a polling function in addition to the interrupt features of CPI configuration 300. In other words, CPI configuration 300 can be configured with only the use of automatic CPU polling as desired, without the use of automatic CPU polling and instead use of only the interrupt features, e.g., TxIRQ and RxIRQ, or with both CPU polling and interrupt features incorporated within.

To further illustrate the operation of an exemplary SPI configuration in accordance with the present invention, an exemplary operation sequence 500 for a buffering scheme is illustrated with reference to Figure 5, and additional reference again to Figures 3 and 4. In accordance with this example, prior to operation, the FIFO buffer 400 is empty and without remaining data to be transmitted or received, and all counters and pointers are configured to the same location, for example, all pointers and counters could be pointing to zero (0000B or 0 decimal). During an initialization step 502, the CPU will write to an SPI data register, for example, SPI data register 314, and the SPI configuration will facilitate the writing of data into the location of FIFO buffer 400 designated by the WrPtr pointer. Thereafter, in a transmit buffering step 504, the WrPtr pointer is suitably incremented to prevent the next byte to be transmitted from overwriting the previous written byte, while the WrShfCnt counter is incremented to keep track of the number of bytes available for transmission For example, in the illustration of Figure 4, with WrPtr equal to 8 decimal, and ShfPtr equal to 3 decimal, WrShfCnt is equal to 5 decimal, indicating 5 bytes are available for writing.

Upon the writing of data to an SPI data register in step 502, and incrementing of the WrPtr pointer and the WrShfCnt counter in step 504, a transmit and receive shifting step 506 can be provided. In step 506, the SPI configuration 300 can begin transmitting the data by reading the data from the ShfPtr location in the buffer, and then writing the

data to the transmit shift register 302. Upon completion of transferring of data from FIFO buffer 308 to the transmit shift register 302, shifting of the registers can suitably occur.

At the same time that transmission of data is being completed, or substantially immediately after transmission of data, for example, within one clock cycle, a new byte of data can be suitably received and stored in the receive shift register 304, i.e., stored in the location of FIFO buffer 400 designated by the RdPtr pointer. Accordingly, by providing a buffering scheme that permits data to be received substantially at the same time, or immediately after, the transmitting of data, a high data rate can be obtained, for example, up to 12 Mhz or more.

After the transmission and receiving of data in steps 504 and 506, a receive buffering step 508 can be suitably provided. In step 508, the ShfPtr pointer can again be incremented, and thus point to a new location in the FIFO buffer 400. In addition, the RdShfCnt can also be suitably incremented to indicate that the SPI configuration 300 has received another byte of data.

As discussed above, the SPI configuration can also be configured with FIFO buffer 308 to suitably interrupt the CPU if data is ready for transmitting, e.g., if the FIFO buffer is nearly full or full of data, or if the FIFO buffer is ready to receive data, e.g., if the buffer is nearly empty or empty. Accordingly, an interrupt step request 510 can be suitably provided. In step 510, if the programmable interrupt request Rx IRQ is set to interrupt the CPU when one byte has been received, then the CPU can suitably read the SPI data register. However, the threshold value of interrupt request RxIRQ could also be configured to delay interrupt of the CPU and have the CPU intervene only when two, three or more bytes have been received. Upon interruption of the CPU, the SPI configuration 300 can then make sure that the CPU can receive the data in the location of FIFO buffer 400 designated by the RdPtr pointer. After the data is received, the RdPtr pointer can be suitably increased, and the RdShfCnt can be decreased, which indicates the amount of new data that is available for the CPU. For example, if the receive interrupt request RxIRQ is set to a threshold value of one, and the CPU reads

the data designated by the RdPtr pointer, the RdShfCnt is decreased to zero, which indicates that no new data is available for the CPU.

While step 510 can be configured to occur after receive buffering step 508, interrupt request step 510 can also be configured to occur during step 508. In addition, interrupt request step 510 can also be configured to occur prior to receive buffering step 508, for example, before or during transmit buffering step 504, or after transmit buffering step 504. Accordingly, interrupt request step 510 can occur any time during the operation of SPI configuration 300 when it is desirable to interrupt the CPU to initiate the transmitting or receiving of data.

Accordingly, FIFO buffer 400 can be configured to essentially overwrite a buffer register which had the data that was transmitted, i.e., the function of FIFO buffer 400 can be configured to essentially transmit and receive data at substantially the same time to provide the throughput of at least, if not greater than, larger sized buffers. In addition, while the above operation provides an example of a single byte being transmitted and received, including the proper incrementing of pointers and counters, and shifting of registers, other variations can be realized. For example, 16 bytes could be written to FIFO buffer 400 prior to any transmission of the data. Moreover, at substantially the same time that, or immediately thereafter, these bytes are being transmitted, 16 bytes could also be suitably received in the FIFO buffer 400 due to the appropriate incrementing and decrementing of the WrPtr pointer and RdPtr pointer. Moreover, any variation up to N bytes, or any variation in between can be realized, for example, four bytes transmitted and four bytes received, or six bytes transmitted, and three bytes received. Such an operation effectively results in the overwriting of data to increase the throughput of the FIFO buffer, e.g., the FIFO buffer is configured to effectively facilitate the sending and receiving of data at the same time.

It should also be noted that the various steps in the exemplary operation sequence 500 can be suitably arranged in alternate ways, and the above embodiment is for illustration purposes only. For example, after transmit buffering step 504, sequence 500 could suitably return to initialization step 502 before proceeding to the transmit and receive shifting step 506. Further, transmit buffering step 504 could be repeated

several times before proceeding to any subsequent step. Moreover, any other like variations of any of the other steps in sequence 500 that are configured to provide high data rates can be implemented.

With reference to Figure 6, an exemplary timing diagram illustrates a data write, transmit and read and transmission sequence. For example, for a case of a FIFO buffer having 16 bytes available, i.e, N = 16, and in which the CPU has written 16 bytes of data to the designated FIFO buffer location, the WrPtr will suitably increment from zero to fifteen (15), with WrPtr remaining at decimal fifteen (15) location since there are no further storage locations. In the event that the CPU writes additional data, that data can also be written to the location designated by WrPtr at a value of decimal fifteen (15). The remaining RdPtr and ShfPtr pointers and the number of bytes available for reading remain at zero during this writing of data. In addition, although not shown, the WrShfCnt counter can also suitably increment from zero to fifteen (15), which would indicate that the FIFO buffer is full. At this point, the SPI is ready to begin transmitting of the written data and receiving of new data.

As the SPI has transmitted the data, the RdPtr pointer will increment as a new byte is received from the CPU. Further, the ShfPtr pointer is also increased to indicate the register for which the newly received byte can be written. At the same time, the WrPtr pointer will suitably decrease for every byte that is transmitted. This process can continue as each of the available bytes are transmitted, and each of the buffer locations have been read by the CPU, until the ShfPtr pointer and RdPtr pointer equal fifteen (15) with the effect that the FIFO buffer can suitably have data transmitted and received at the same time. At this point, or at any other desired time when the SPI desires to have data read by the CPU, the SPI could interrupt the CPU to have the CPU to begin reading the data. As a result, the SPI is selectively configured to determine how many bytes are to be read, to be written and to be sent, as well as how many of each category should be stored in FIFO buffer before the CPU is interrupted.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Rejection under 35 USC § 103(a) as being unpatentable over U.S. Patent No. 5,278,956 in view of U.S. Patent Application Pub. No. US 2002/0078317 A1.

ARGUMENT

Rejection under 35 USC § 103(a) over U.S. Patent No. 5,278,956 in view of U.S. Patent Application Pub. No. US 2002/0078317 A1.

Claims 15-17, 19, and 20

Claim 15 includes "... initializing a single buffer to act as transmitter and receiver by writing data to a data register; ... performing a transmit and receive shifting sequence to facilitate transmitting of the data and receiving of new data at substantially the same time ...". The references of record do not show, teach, or suggest the above recited limitations of claim 15. U.S. Patent Application Pub. No. US 2002/0078317 A1 does not disclose a single buffer to facilitate transmitting and receiving at substantially the same time. The references do not teach how the device of U.S. Patent Application Pub. No. US 2002/0078317 A1 can be combined with U.S. Patent No. 5,278,956 to obtain a single buffer for performing a transmit and receive shifting sequence to facilitate transmitting of the data and receiving of new data.

CONCLUSION

For the foregoing reasons, Appellants respectfully submit that the Examiner's final rejection of Claims 15-17, 19, and 20 is improper, and it is respectfully requested that the Board of Patent Appeals and Interferences so find and reverse the Examiner's rejection.

Please charge any fees necessary in connection with the filing of this paper, including any necessary extension of time fees, to Deposit Account No. 20-0668 of Texas Instruments Incorporated.

Respectfully submitted,

/Alan K. Stewart/
Alan K. Stewart
Registration No. 35,373
Attorney for Appellant

Texas Instruments, Incorporated
P. O. Box 655474 - M/S 3999
Patent Department
Dallas, Texas 75265
972/917-5466

CLAIMS APPENDIX

15. A high performance buffering technique for use with a serial peripheral interface to facilitate high data rates, said buffering technique comprising the steps of

initializing a single buffer to act as transmitter and receiver by writing data to a data register;

performing a transmit buffering sequence to prepare for the transmitting of the data;

performing a transmit and receive shifting sequence to facilitate transmitting of the data and receiving of new data at substantially the same time; and

performing a receive buffering sequence to prepare for the receipt of additional new data.

16. The high performance buffering technique of claim 15, wherein said initialization step comprises the step of:

writing the data into a location of said buffer as designated by a write pointer.

17. The high performance buffering technique of claim 15, wherein said transmit buffering sequence comprises the steps of:

incrementing a write pointer to prevent a next byte to be transmitted from overwriting a previous written byte; and

incrementing a write shift counter to facilitate tracking of a number of bytes available for transmission.

19. The high performance buffering technique of claim 15, wherein said receive buffering sequence comprises the steps of:

incrementing a shift pointer to identify a new location in the buffer for receiving data; and

incrementing a read shift counter to indicate that the new data has been received.

20. The high performance buffering technique of claim 15, wherein said buffering technique further comprises the steps of:

interrupting a CPU if the data is ready for transmitting and said buffer is approximately full; and

interrupting the CPU if said buffer is ready to receive data and said buffer is approximately empty.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.